

# kantox

Tomorrow's FX today

**Milad Zaghab.**

**Kantox's QA Job application.**

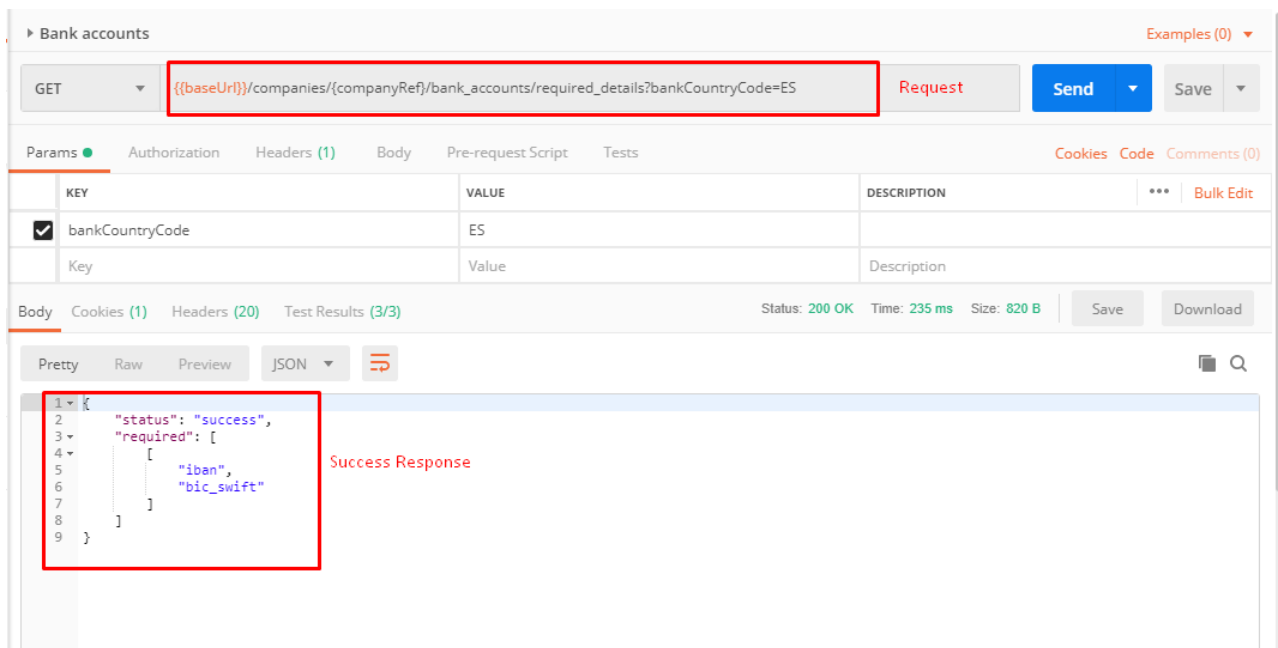
*"Quality means doing things well even when nobody is watching"*

Henry Ford.

## Kantox QA Exercise

The first part of the exercise, the exploratory session, was approached in two ways; the first one, was about getting to know the business rules and how the API was programmed, as well as looking at which details were required and what they mean in the context.

For the second approach, after studying some tools, I decided to work with Postman, it allowed me to get familiar with the API in an easy way. After studying the documentation provided by Kantox, a Test Case for the session was written, which can be find in the spreadsheet attached. To test it using Postman the exploratory test session took around 45 minutes, with the chosen tool it was possible to automate the token authorization process, to make some request and analyze the response, to organize and divide the work by environments and folders as well as providing unique data for each environment. Additionally, with the tool it was easy to change parameters, read and interpret the responses, evaluate the errors when a required data was not provided or when for example, wrong data type was provided.



Img 1. Bank Account request and response.

Filter

History Collections

Trash

Kantox  
6 requests

Kantox Collections

Kantox Test  
7 requests

Valid input

Login

Organization by folders

Get counter value

Create order

POST Create order

Request

Invalid input

POST Login

POST Get counter value

POST Create order

KantoxEnvironment

Kantox Environment

Examples (0)

POST {{baseUrl}}/companies/{{companyRef}}/orders/quote

Send Save

Params Authorization Headers (1) Body Pre-request Script Tests

none form-data x-www-form-urlencoded raw binary

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> marketDirection	{{marketDirection}}	
<input checked="" type="checkbox"/> currency	{{currency}}	
<input checked="" type="checkbox"/> amount	{{amount}}	
<input checked="" type="checkbox"/> counterCurrency	{{counterCurrency}}	
<input checked="" type="checkbox"/> valueDate	{{valueDate}}	

Key Text Value Variables to read from the file Description

Img. 2 File organization and variable parametrization.

## **Test Case Automation**

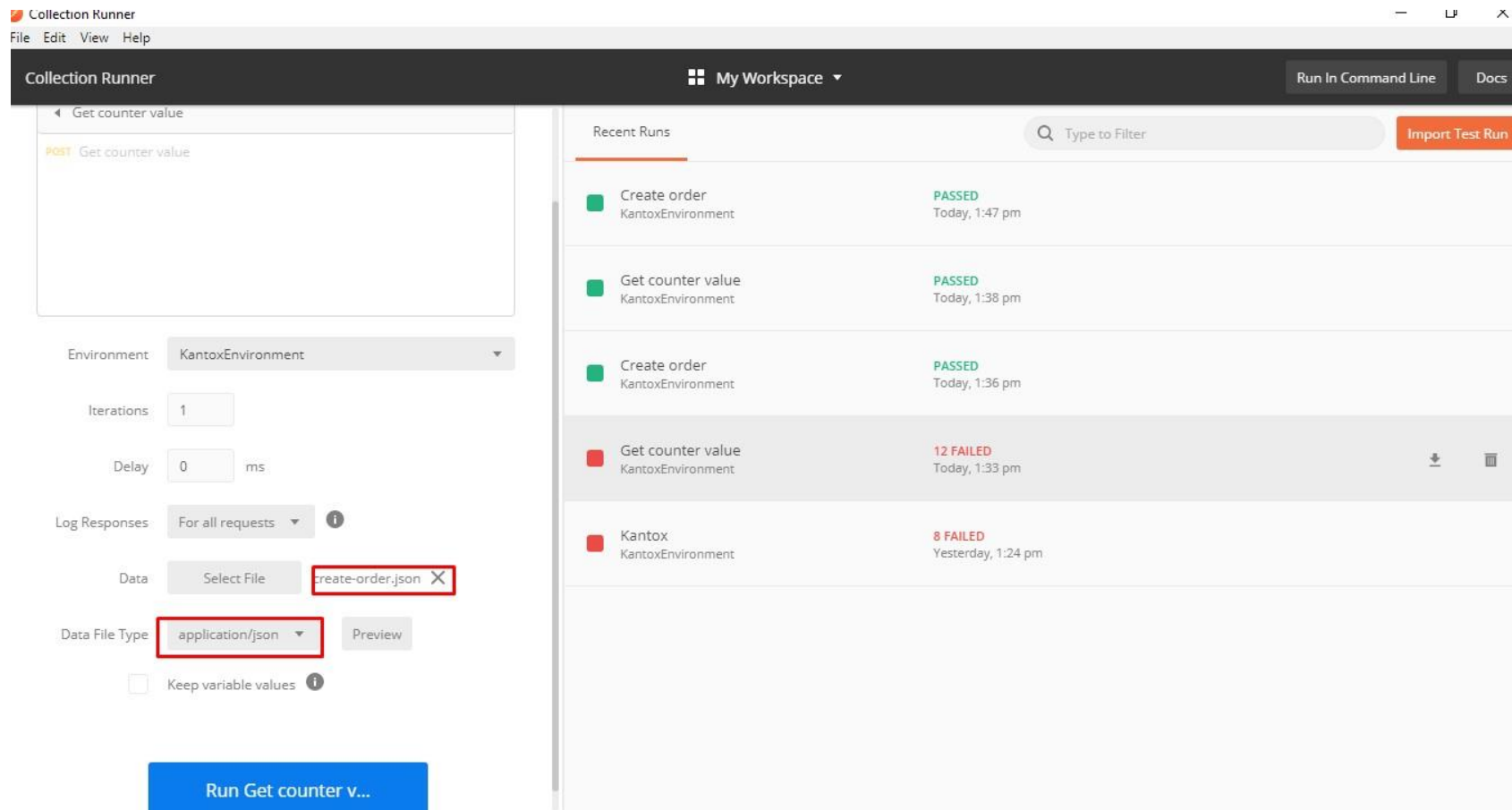
For the second section, test cases were written for the order's management. The first one intention was to quote an order, the second one to create an order. The strategy used to test this section of the API was to create a Postman collection that allowed me to organize the requests in two folders, valid and invalid Data.

The parametrization for the requests was made with Postman collection variables, and from a data file. Collection variables were useful for testing common values used throughout requests, like the token, company reference, beneficiary account reference, login credentials, etc.

On the other hand, values from the data file offered a central, reusable and maintainable way to test the behavior of the endpoint on each request. It was used to assert the correct creation of the order with different values. To this end, multiple tests were written to check the response of the server. Things like the value of the status code, presence and non-emptiness of some fields were heavily tested to check that everything was ok.

Similarly, the behavior of the API with invalid data was also tested, checking that the appropriate error codes and fields were returned from the server.

All in all, using Postman allowed a familiar, simple and maintainable way of testing the API and generating useful test runs summaries (with the collection runner); also, it allows to share the whole collection (with its variables) to another member of a team, so everyone can have the same base for testing.



Img. 3 Json file upload.

EDIT COLLECTION

Name

Kantox Test

Description

Authorization

Pre-request Scripts

Tests

Variables

These variables are specific to this collection and its requests. [Learn more about collection variables.](#)

	VARIABLE	INITIAL VALUE ⓘ	CURRENT VALUE ⓘ	...	Persist All	Reset All
<input checked="" type="checkbox"/>	baseUrl	https://demo.kantox.co...	https://demo.kantox.com/api			
<input checked="" type="checkbox"/>	companyRef	1SZKCL2H1	1SZKCL2H1			
<input checked="" type="checkbox"/>	beneficiaryAccountRef	BA-26Q91JZ9P	BA-26Q91JZ9P			
<input checked="" type="checkbox"/>	login	apiusertest@kantox.com	apiusertest@kantox.com			
<input checked="" type="checkbox"/>	password	!2018abde123	!2018abde123			
	Add a new variable					

ⓘ

 Use variables to reuse values in different places. Work with the current value of a variable to prevent sharing sensitive values with your team. [Learn more about variable values](#)

Cancel

Update

Img. 4 Collection's variables.

Collection Runner

File Edit View Help

Collection Runner Run Results My Workspace Run In Command Line Docs

16 PASSED 0 FAILED Get counter value KantoxEnvironment just now

Run Summary Export Results Retry New

Iteration 1

POST Get counter value https://demo.kantox.co... ... counter value / Get counter value 200 OK 1698 ms 256 B

- PASS Status code is 200
- PASS Status value is success
- PASS Response body should not be empty
- PASS Should have ratePair property
- PASS marketDirection should not be empty
- PASS marketDirection value should be equal to the provided on input
- PASS ratePair should not be empty
- PASS ratePair value should concatenate currency and CounterCurrency
- PASS Should have rate property
- PASS rate should not be empty
- PASS Should have marketDirection property
- PASS Should have amount property

Img. 5. Test results examples.